УДК 004.432.4

РАСШИРЕНИЕ СИСТЕМЫ ПРОГРАММИРОВАНИЯ ТУРБО ПРОЛОГ ПРЕДИКАТАМИ ДЛЯ РАБОТЫ СО СПИСКАМИ

Канд. техн. наук, доц. КОВАЛЬКОВ А. Т., инж. КОВАЛЬКОВА И. А.

Белорусский национальный технический университет

Язык логического программирования Пролог разработан в начале 70-х гг. XX в. в Европе как один из языков для решения задач, связанных с проблематикой искусственного интеллекта. В последующие годы появилось много версий языка, среди которых наиболее популярной является Турбо Пролог [1]. Интерес к Прологу поднимался и затихал несколько раз. В 1980-е гг. он был выбран в качестве базового языка в японском проекте компьютеров пятого поколения. В настоящее время Пролог продолжает развиваться и вбирает в себя новые технологии и концепции программирования [2]. Показательным является пример использования Пролога в одном из самых амбициозных и успешных проектов в ВПК России – «Буран». На сайте [2] имеются десятки и сотни тысяч ссылок на публикации по реализации и применению Пролога. К сожалению, публикаций на русском языке среди них очень мало. В последние годы появилось несколько версий (5.2, 6.1, 6.3, 7.0, 7.1) визуальной среды программирования Visual Prolog [3], реализующей Пролог на качественно новом уровне.

Турбо Пролог имеет достаточно большой набор встроенных предикатов (аналогом предиката в других языках является процедура) для работы со строками, файлами, окнами, базами данных, для организации ввода-вывода данных и др. Однако Турбо Пролог не имеет стандартных предикатов для работы со списками, которые являются основной структурой данных в этом языке программирования. Для каждой из операций со списками приходится писать отдельные правила, что отнимает значительную часть времени программиста при разработке программного обеспечения на языке Турбо Пролог.

Кроме того, при работе со строками часто приходится преобразовывать их в списки символов или атомов (слов), а при работе с текстовыми файлами – в списки строк. Поэтому наличие стандартных предикатов для выполнения таких часто встречающихся операций со списками, как ввод списка, проверка принадлежности элемента списку, объединение списков, выделение первого (последнего) элемента списка, удаление элементов из списка, сортировка, нахождение максимального (минимального) элементов списка и т. д., намного бы облегчило труд программиста.

В данной работе описывается система программирования Турбо Пролог, дополненная встроенными предикатами для работы со списками.

С помощью встроенных предикатов можно выполнять следующие операции со списками:

- ввод списка (количество элементов списка может задаваться или не задаваться);
- выделение элементов списка (первого, последнего, максимального, минимального);
- удаление элементов списка (первого, последнего, первого вхождения или всех вхождений заданного элемента, повторяющихся элементов, т. е. преобразование списка в множество);
- добавление элементов (в начало или конец списка, вставка элемента в отсортированный список);
- сортировку списков (в порядке возрастания или убывания элементов, а также в алфавитном или обратном алфавитном порядке);
- другие операции (принадлежность элемента списку, объединение списков, реверсирование списка, подсчет суммы или количества элементов списка).

Все эти операции могут выполняться над пятью типами данных: integer, real, char, string, symbol. Используя допускаемое Турбо Прологом описание предикатов и определенную последовательность их расположения в разделе predicates, удалось добиться того, что вместо пяти вариантов одного и того же правила для каждого из возможных типов элементов списка в разделе clauses достаточно было записать только два варианта: один для типов integer, real, string, symbol, второй – для типа char.

Все правила располагаются в одном модуле, который подсоединяется к разрабатываемой

программе с помощью директивы компилятора include, например,

include “list.pro”,

где list.pro – имя файла модуля.

Стандартная справка о встроенных предикатах, вызываемая клавишей F1, дополнена разделом «Предикаты для работы со списками». Как и для имеющихся стандартных предикатов Турбо Пролога, так и для новых предикатов для работы со списками, краткая справка по выбранному предикату имеет формат:

имя_предиката (Перечень_параметров) (типы_параметров) : (поточный шаблон) – описание_предиката.

В поточном шаблоне указываются входные (i) и выходные (o) параметры, в описании предиката определяются выполняемая им операция, а также назначение каждого из параметров.

Более подробную справку о предикатах по работе со списками можно получить из автономного справочного файла, после запуска которого появляется меню, позволяющее выбрать нужный предикат и получить по нему справку. Эта справка может быть вызвана как со среды ДОС, так и со среды Турбо Пролога через временный выход в ДОС с последующим возвратом в среду программирования по команде exit.

Ниже представлены два листинга программы построения полиндрома. Полиндром – это список, который читается одинаково как с начала, так и с конца, например, [1, 2, 3, 4, 5, 4, 3, 2, 1]. Для его построения нужно выполнить следующую последовательность операций над исходным списком: ввод списка, удаление из списка последнего элемента, реверсирование исходного списка, объединение списка без последнего элемента с реверсированным списком.

Листинг 1. Программа построения полиндрома.

```
domains
  l=real*
predicates
  read_l_real(l)
  del_last_el(l,l)
  reverse(l,l)
  concat_l(l,l,l)
clauses
  % Ввод списка, содержащего произвольное число элементов типа real
  read_l_real([H|T]):-
readreal(H),!,read_l_real(T).
  read_l_real([]).
  % Удаление последнего элемента списка
  del_last_el([_],[]).
  del_last_el([H|T],[H|T1]):-del_last_el(T,T1).
  % Обращение списка
  reverse([],[]).
  reverse([H|T],Lrev):-reverse(T,Trev),
                       concat_l(Trev,[H],Lrev).
  % Объединение двух списков
  concat_l([],L,L).
  concat_l([H|T],L,[H|T1]):-concat_l(T,L,T1).
goal
  clearwindow, write(“Введите список :”),nl,
  read_l_real(L),del_last_el(L,L1),
  reverse(L1,Lrev),concat(L,Lrev,P),
  write(“L=”,L),nl,write(“L1=”,P).
```

Листинг 2. Программа построения полиндрома с подключенным модулем, содержащем предикаты для работы со списками.

```
include “list.pro”
goal
  clearwindow, write(“Введите список :”),nl,
  read_l_real(L),del_last_el(L,L1),
  reverse(L1,Lrev),concat(L,Lrev,P),
  write(“Исходный список = ”,L),nl,write
(“Полиндром = ”,P).
```

Результаты работы применимы и в системе визуального программирования Visual Prolog.

В Ы В О Д

Турбо Пролог, дополненный встроенными предикатами для работы со списками, имеет ряд достоинств:

- значительно сокращает время на разработку программы;
- объем листинга программы сокращается в несколько раз за счет того, что в нем нет правил, которые содержатся в подключаемом модуле;
- система программирования и справочная система легко расширяемы новыми встроенными предикатами;
- при достаточном количестве встроенных предикатов (не только для работы со списками) для решения задач определенного класса программа может состоять из одного раздела goal (цель), и эту программу способен написать пользователь, имеющий минимальные навыки в программировании на языке Турбо Пролог.

Л И Т Е Р А Т У Р А

1. **Шрайнер, П. А.** Основы программирования на языке Пролог / П. А. Шрайнер. – М.: Интернет-ун-т информ. технологий, 2005. – 176 с.

2. **http://ru.wikipedia.org/wiki/Prolog**

3. **Адаменко, А. Н.** Логическое программирование и Visual Prolog / А. Н. Адаменко, А. М. Кучуков. – СПб.: БХВ-Петербург, 2003. – 992 с.

Поступила 2.02.2007